Getting started with Docker + Wasm

Nigel Poulton(https://nigelpoulton.com/author/ashpoulton/) November 8, 2022(https://nigelpoulton.com/2022/11/08/)

Get your hands-on with Docker and Wasm - write an app, compile it to Wasm, package it as an OCI image, store it in Docker Hub, run it with Docker...

UPDATED: 26th September 2023.

Docker is constantly adding and improving support for WebAssembly in **Docker Desktop (https://www.docker.com/products/docker-desktop/)**. They also announced **Docker Hub support for WebAssembly artifacts (https://www.docker.com/blog/announcing-docker-hub-oci-artifacts-support/)**.

As I'm all over WebAssembly, I thought I'd take both of them for a spin and write a blog post to help anyone else wanting to get hands-on with Docker and Wasm.

The Big Picture

WebAssembly (also known as Wasm) creates very small, very fast binaries that execute in a secure sandbox anywhere that has a WebAssembly runtime. To be clear... they're **a lot smaller, a lot faster, a lot more secure**, and **a lot more portable** than containers. And that's coming from me — a container fanboy.

As a quick example of size... Michael Yuan recently **shared in a tweet (https://twitter.com/juntao/status/1588528265953652737)** that he has a full HTTP server + MySQL client "app server" that's only 2MB!. Others have shared even smaller examples.

Anyway, **WebAssembly is becoming a big deal in the cloud native (https://nigelpoulton.com/webassembly-the-future-of-cloud-computing/)** space and Docker wants a piece of the action.

Hands-on with Docker and Wasm

I'm about to walk you through writing a *hello world* Wasm app, packaging it as an OCI image, pushing it to Docker Hub, and running it with Docker. It's super-simple, but you'll need both of the following if you want to follow along.

- **Docker Desktop (https://www.docker.com/products/docker-desktop/)**
- **Rust (https://www.rust-lang.org/tools/install)** programming language

At the time of writing, you'll need to enable the ***Use containerd for pulling and storing images*** and ***Enable Wasm*** checkboxes on the Settings tab of Docker Desktop.

With Docker Desktop and Rust installed, we'll complete these steps:

1. Configure Rust to compile code to Wasm
2. Write the app
3. Compile the app into a Wasm binary
4. Package the app into an OCI image
5. Push the image to Docker Hub
6. Run the Wasm app using Docker

Perform all of the following tasks from a command line.

Configure Rust to compile code to Wasm

Run the following `rustup` command to install the **wasm32-wasi target (https://doc.rust-lang.org/rustc/targets/index.html#:~:text=rustc%20is%20a%20cross%2Dcompiler,target%2C%20see%20the%20docs** so that Rust can compile source code into Wasm binaries.

```
rustup target add wasm32-wasi
```

If you run a `rustup target list` and **wasm32-wasi (installed)** appears in the list, Rust is configured and you're ready to create the app.

Write the app

Run the following command to scaffold up a simple Rust app. It'll create a simple Hello World app in `hello-docker/src/main.rs`.

```
cargo new hello-docker
```

Feel free to inspect the `hello-docker/src/main.rs` file and change the text printed to the screen. I've changed mine to print "Hello, Docker Hub!" as follows.

```
fn main() {
    println!("Hello, Docker Hub!");
}
```

At this point the Rust app is written and ready to be compiled as a Wasm binary.

Compile the app into a Wasm binary

Change into the `hello-docker` directory and run the following `cargo` command to compile the Rust app as a **wasm32-wasi** binary. This will create a Wasm bytecode binary that will run on any system with a WebAssembly runtime.

The `cargo` command is installed as part of Rust. It also knows where to find the source code and how to compile it as a Wasm binary.

```
cd hello-docker
cargo build --target wasm32-wasi --release
```

The command outputs the compiled `hello-docker.wasm` Wasm binary into the `hello-docker/target/wasm32-wasi/release` folder.

We'll execute it in an upcoming step with the help of Docker. However, before doing that, we'll build it into an OCI image so it can be stored in Docker Hub and executed by Docker.

Build a Wasm app into an OCI image

Docker can package a Wasm module into an OCI image (that's just a fancy name for a Docker image).

The way to do this right now feels a bit hacky and I expect it to change in the future. However, you start with a **scratch base image (https://support.snyk.io/hc/en-us/articles/360004012857-What-are-docker-scratch-based-images-#:~:text=The%20scratch%20image%20is%20the,statically%20compiled%20and%20self%2Dcontained.)**, copy in the Wasm module, and set the program to execute as the Wasm binary.

The following `Dockerfile` describes this. Create it in your current directory (you should be in your `hello-docker` directory).

```
FROM scratch
COPY ./target/wasm32-wasi/release/hello-docker.wasm /hello-docker.wasm
ENTRYPOINT [ "hello-docker.wasm" ]
```

With the **Dockerfile (https://docs.docker.com/engine/reference/builder/)** created, run the following command to build the image. The command assumes you're in the same directory as the Dockerfile.

```
docker buildx build --platform wasi/wasm --provenance=false -t docker-
wasm:0.1 .
```

The `--platform wasi/wasm` flag sets the target OS for the image to **wasi** and target architecture to **wasm**. The `-t docker-wasm:0.1` tags the image "docker-wasm:0.1" and the period at the end tells Docker to use the Dockerfile in the current directory.

Run the following command to verify the new image exists.

```
docker images

REPOSITORY      TAG    IMAGE ID        CREATED   SIZE
docker-wasm     0.1    6b43e0bdf164    2 mins    501kB
```

At this point the Wasm app is packaged in an OCI image.

Push the Wasm app to Docker Hub

This is an optional step. Skip it if you don't care about using Docker Hub. However, if you're following this step, you'll need a **Docker ID (https://hub.docker.com/signup)**. They're free, and they're important if you're serious about learning and working with Docker.

Run the following command to tag the image with *your own Docker ID* so you can push it to your own Docker Hub repositories. My Docker ID is `nigelpoulton`, so I'll run the command below.

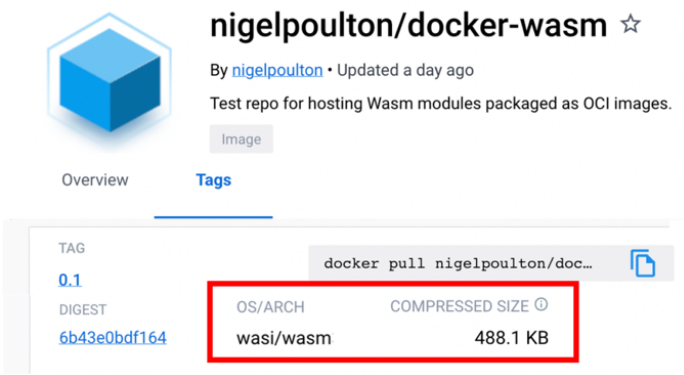
```
docker tag docker-wasm:0.1 nigelpoulton/docker-wasm:0.1
```

With the image tagged with your Docker ID, run the following command to push it to Docker Hub. Remember to substitute your own Docker ID. This failed the first time I tried, and I had to perform a manual `docker login` and run the command again.

```
docker push nigelpoulton/docker-wasm:0.1

6b43e0bdf164: Pushed
33b9d7fa88a0: Pushed
4c601df9af6e: Pushed
0.1: digest: sha256:6b43...31f8, size: 526
```

At this point the Wasm app is packaged as an OCI image and hosted on Docker Hub. Note the `OS/ARCH` and `SIZE` fields.



Run the Wasm app using Docker

You'll need Docker Desktop version 4.15 or later to complete this step.

Run the following command to tell Docker to run the Wasm app packaged in the OCI image. It's a single command wrapped over multiple lines.

```
docker run --rm --name=dockerwasm \
  --runtime=io.containerd.wasmedge.v1 \
  --platform=wasi/wasm \
  nigelpoulton/docker-wasm:0.1
```

The `--runtime=io.containerd.wasmedge.v1` flag is how Docker tells **containerd (https://containerd.io/)** to use the **runwasi containerd shim (https://github.com/containerd/runwasi)** and call **WasmEdge (https://wasmedge.org/)** to run the Wasm module packaged in the image. More on this works **in this post (https://nigelpoulton.com/what-is-runwasi/)**.

This will be the output.

```
Hello, Docker Hub!
```

Congratulations. You just used Docker to build, share and run a Wasm application!

Adding support for more runtimes

Docker is constantly adding features and improving support for Wasm in Docker Desktop. The latest tech preview release (4.24) supports the following Wasm runtimes:

- **wasmtime (https://wasmtime.dev/)**
- **WasmEdge (https://wasmedge.org/)**
- **Spin (https://www.fermyon.com/spin)**
- **slight (https://github.com/deislabs/spiderlightning)**
- **wasmer (https://wasmer.io/)**
- **lunatic (https://lunatic.solutions/)**
- **wws (https://github.com/vmware-labs/wasm-workers-server)**

Conclusion

It's very early and some of the stuff is a bit hacky at the time I'm writing. However, the direction is clear and **it's game-changing to be able to use familiar Docker tools and skills to work with WebAssembly apps**.

This is also just the tip of the iceberg with Docker and Wasm!

Other WebAssembly/Wasm articles

- **WebAssembly: The future of cloud computing (https://nigelpoulton.com/webassembly-the-future-of-cloud-computing/)**
- **What is cloud native WebAssembly (https://nigelpoulton.com/what-is-cloud-native-webassembly/)**
- **What is runwasi (https://nigelpoulton.com/what-is-runwasi/)**
- **Docker Hub and WebAssembly – step 1 (https://nigelpoulton.com/docker-hub-and-webassembly-step-1/)**

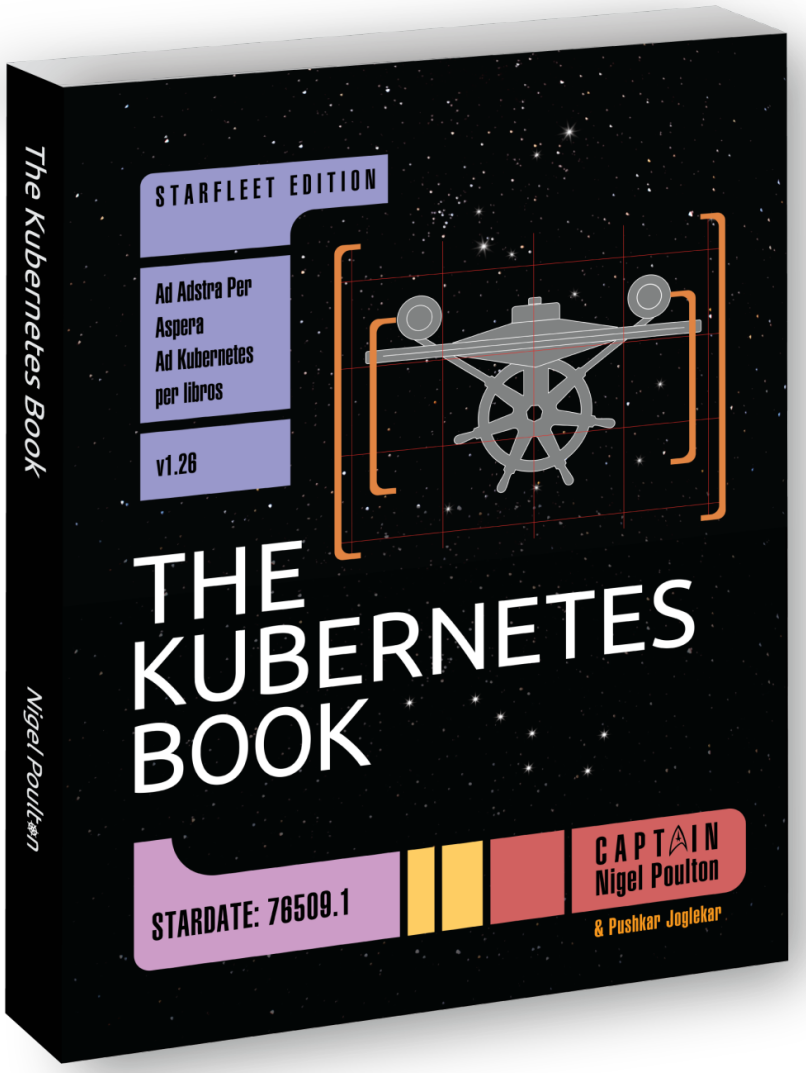
You can also subscribe to my **Word on the cloud newsletter (https://www.getrevue.co/profile/nigelpoulton)**. It's short and keeps you up-to-date with the best stuff going on around cloud native.

Share this post

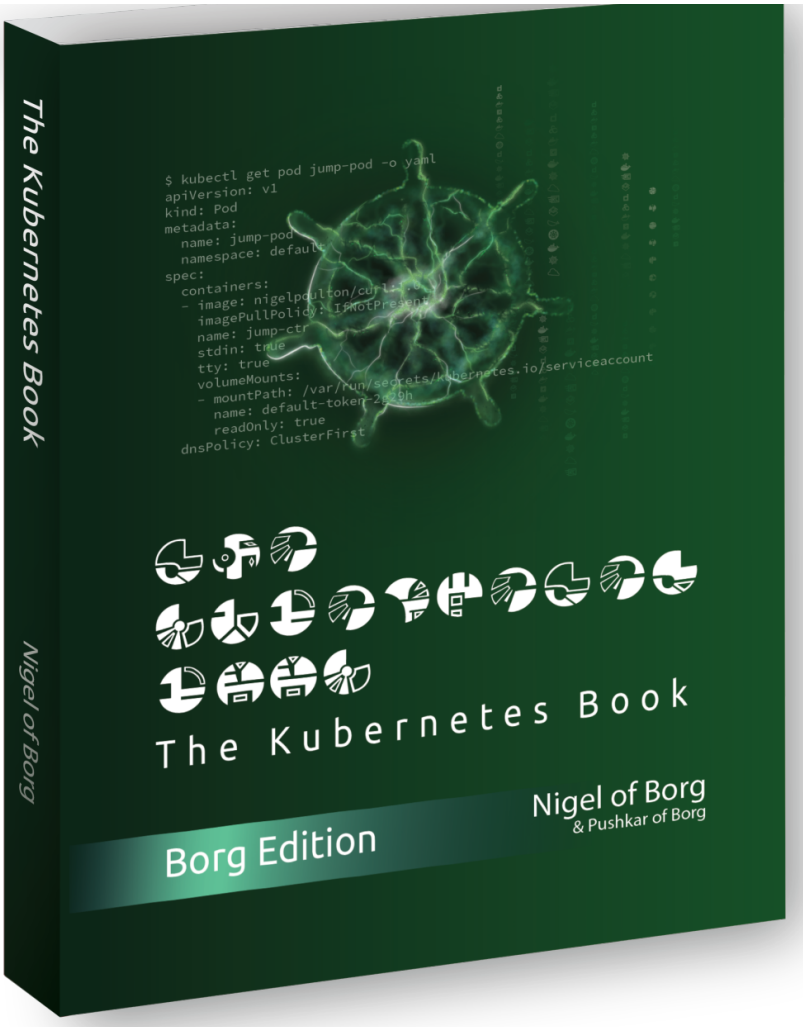
(https://nigelpoulton.com/books/)

Learn More
(https://nigelpoulton.com/books/)

Special Editions



(https://www.amazon.com/Kubernetes-Book-Starfleet-Nigel-Poulton/dp/B0BV1PRJ92/ref=tmm_pap_swatch_0?_encoding=UTF8&qid=1676465946&sr=8-1)



(https://www.amazon.com/

Book-Borg-Collectors/dp/B0BTKZW329/ref=sr_1_1?crid=2EFUWY9OE2N6&keywords=the+kubernetes+book+Borg&qid=1676023771&s=audible&sprefix=the+kubernetes+book+b%2Caudible-1-catcorr)

(https://www.amazon.com/Kubernetes-Book-Klingon-Nigel-Poulton/dp/B0BTKVMF55/ref=sr_1_1?crid=2B9G2ZY0BPE4F&keywords=The+Kubernetes+Book+Klingon&qid=1676023527&sprefix=the+kubernetes+book+klingon%2Caps%2C21)

(/)

Quick Links

Video Courses

Books

Training & Events

About

Privacy Policy

Blog

Vendor Booth Support

K8's Trivia

YouTube Channel

Merch & Charities

(https://twitter.com/nigelpoulton) (https://hachyderm.io/@nigelpoulton) (https://www.youtube.com/@NigelPoulton) (https://www.instagram.com/nigel.poulton/) (https://www.facebook.com/NigelPoultonTrainer/) (https://www.linkedin.com/in/nigelpoulton/)

Follow Nigel

Contact

Name

Email

Message

Send

Subscribe

Word on the cloud: What's going on in cloud native

Nigel's Keeping you up-to-date on cloud native. Short & sharp! #Docker #Kubernetes #WebAssembly #Wasm

Click to subscribe (https://nigelpoulton.substack.com/)